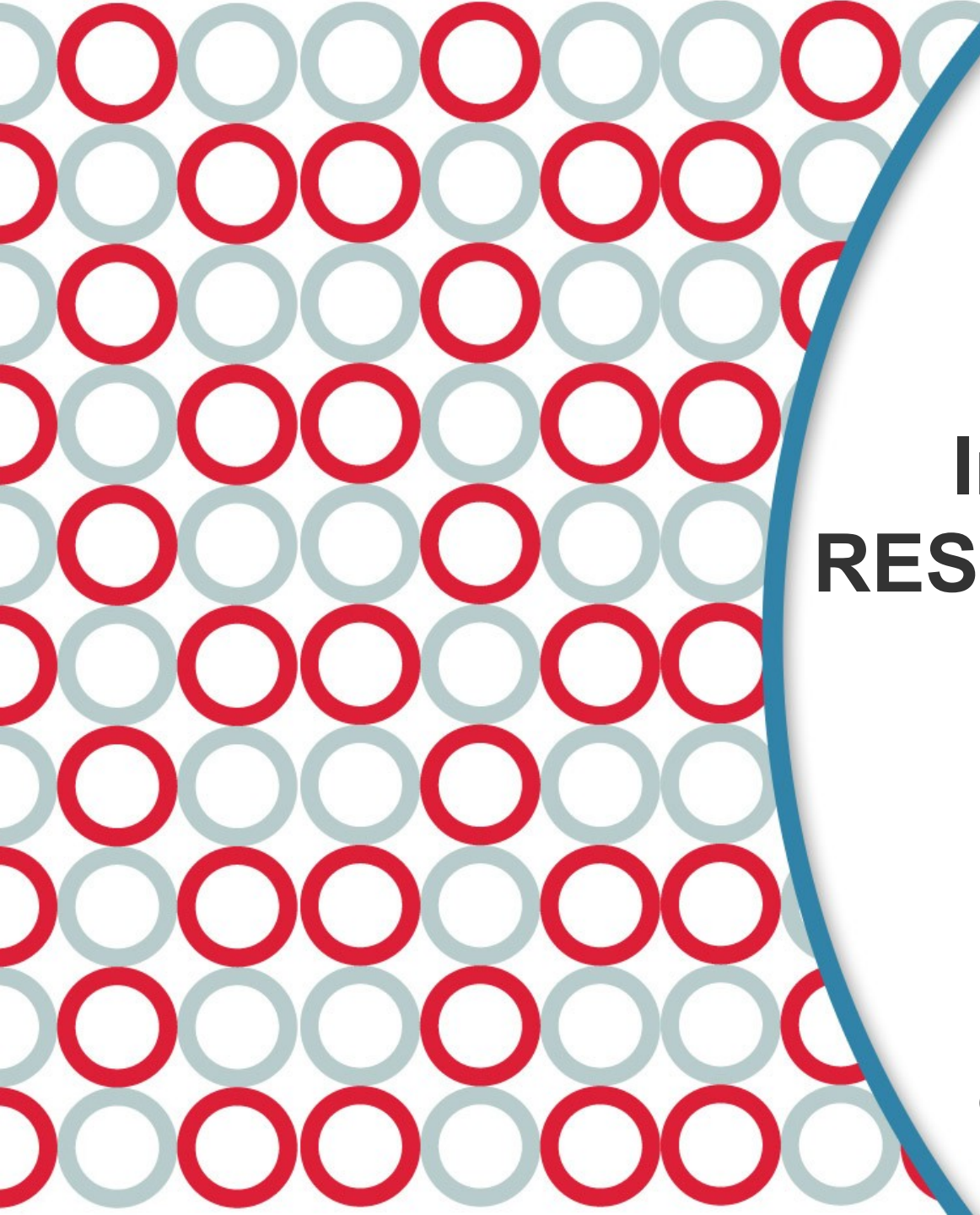


September 2022,
Introduction to Spring 5

Introduction to REST & HATEOAS

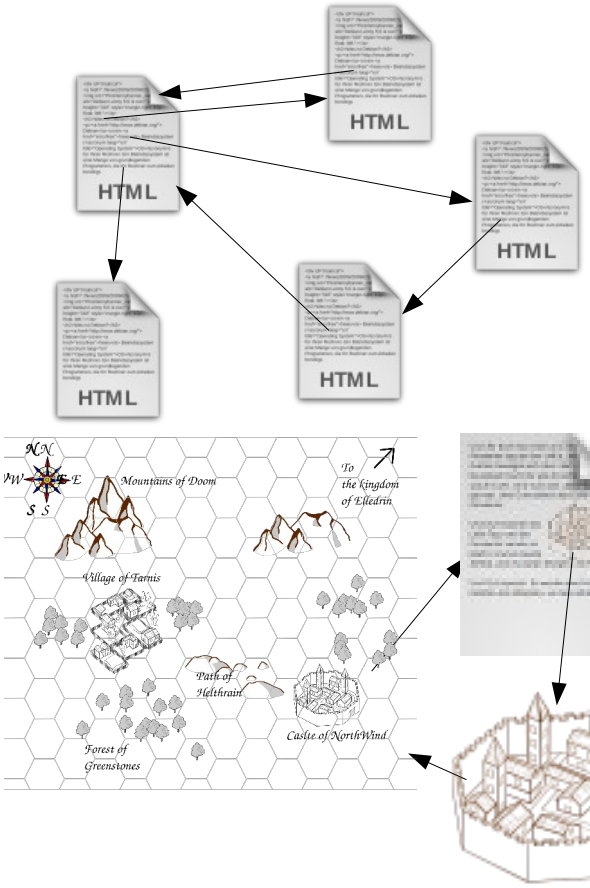
Trayan Iliev
tiliev@iproduct.org
http://iproduct.org

Agenda for This Session

- Multimedia and Hypermedia – basic concepts
- Service Oriented Architecture (SOA),
- Cloud computing and client side mashups
- REpresentational State Transfer (REST)
- Hypermedia As The Engine Of Application State (HATEOAS)
- New Link HTTP header
- Richardson Maturity Model of Web Applications
- Cross Origin Resource Sharing

Hypertext & Hypermedia

- Hypertext is structured text that uses logical links (hyperlinks) between nodes containing text
- HTTP is the protocol to exchange or transfer hypertext
- Hypermedia - extension of the term hypertext, is a nonlinear medium of information which includes multimedia (text, graphics, audio, video, etc.) and hyperlinks of different media types (e.g. image or animation/video fragment can be linked to a detailed description.



Multipurpose Internet Mail Extensions (MIME)

- Different types of media are represented using different text/binary encoding formats – for example:
  - Text -> plain, html, xml ...
  - Image (Graphics) -> gif, png, jpeg, svg ...

- Multipurpose Internet Mail Extensions (MIME) allows the client to recognize how to handle/present the particular multimedia asset/node:

  Media Type Media SubType (format)

  Ex.: Content-Type: text/plain

- More examples for standard MIME types:
  https://developer.mozilla.org/en-US/docs/Web/HTTP/Basics_of_HTTP/MIME_types

- Vendor specific media (MIME) types: application/vnd.*+json/xml

HTTP Request Structure

GET */context/Servlet*
HTTP/1.1
Host:
Client_Host_Name
Header2: *Header2_Data*
...
HeaderN:
HeaderN_Data
<Празен ред>

POST */context/Servlet*
HTTP/1.1
Host: *Client_Host_Name*
Header2: *Header2_Data*
...
HeaderN: *HeaderN_Data*
<Празен ред>
POST_Data

HTTP Response Structure

HTTP/1.1 200 OK

Content-Type:
application/json

Header2: *Header2_Data*

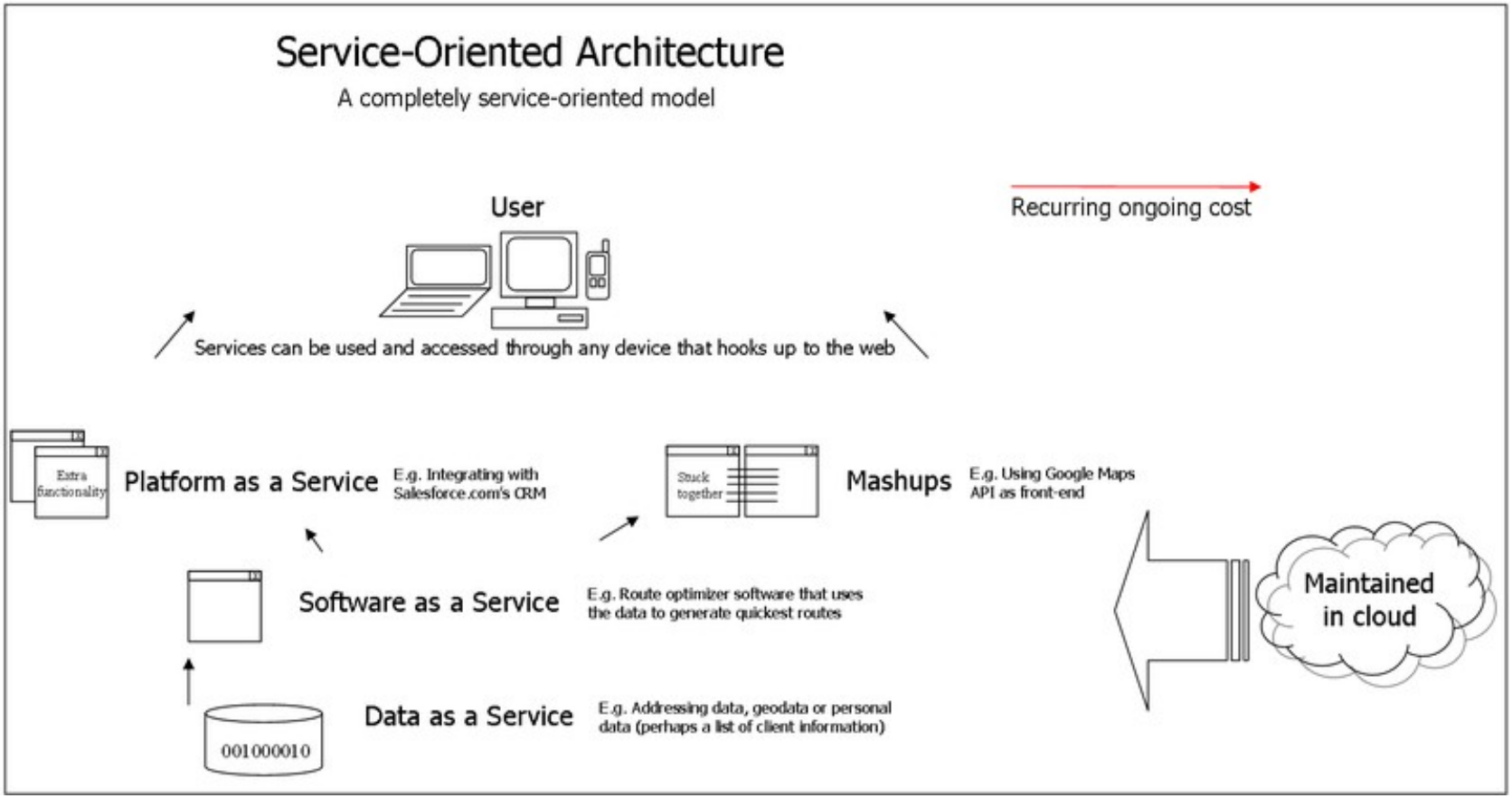
…

HeaderN: *HeaderN_Data*

<Празен ред>

```
[{ "id":1,
   "name":"Novelties in Java EE 7 ...",
   "description":"The presentation is ...",
   "created":"2014-05-10T12:37:59",
   "modified":"2014-05-10T13:50:02",
 },
 { "id":2,
   "name":"Mobile Apps with HTML5 ...",
   "description":"Building Mobile ...",
   "created":"2014-05-10T12:40:01",
   "modified":"2014-05-10T12:40:01",
 }]
```

Service Oriented Architecture (SOA)



Architectural Properties

According to **Dr. Roy Fielding** [Architectural Styles and the Design of Network-based Software Architectures, 2000]:

–Performance

–Scalability

–Reliability

–Simplicity

–Extensibility

– Dynamic evolvability

– Cusomizability

– Configurability

– Visibility

❖All of them should be present in a desired Web Architecture and REST architectural style tries to preserve them by consistently applying several **architectural constraints**

REST Architecture

According to **Roy Fielding** [Architectural Styles and the Design of Network-based Software Architectures, 2000]:

- Client-Server
- Stateless
- Uniform Interface:
  - Identification of resources
  - Manipulation of resources through representations
  - Self-descriptive messages
  - Hypermedia as the engine of application state (HATEOAS)
- Layered System
- Code on Demand (optional)

Representational State Transfer(REST)

- REpresentational State Transfer (REST) is an architecture for accessing distributed hypermedia web-services
- The resources are identified by URIs and are accessed and manipulated using an HHTP interface base methods (GET, POST, PUT, DELETE, OPTIONS, HEAD, PATCH)
- Information is exchanged using representations of these resources
- Lightweight alternative to SOAP+WSDL -> HTTP + Any representation format (e.g. JavaScript™ Object Notation – JSON)

Representational State Transfer(REST)

- Identification of resources – URIs
- Representation of resources – e.g. HTML, XML, JSON, etc.
- Manipulation of resources through these representations
- Self-descriptive messages - Internet media type (MIME type) provides enough information to describe how to process the message. Responses also explicitly indicate their cacheability.
- Hypermedia as the engine of application state (aka **HATEOAS**)
- Application contracts are expressed as **media types** and [semantic] link realtions (**rel** attribute - RFC5988, "Web Linking")

Source: http://en.wikipedia.org/wiki/Representational_state_transfer

Hypermedia As The Engine Of Application State (HATEOAS) – New Link Header (RFC 8288) Example

```
Content-Length →1656
Content-Type →application/json
Link →<http://localhost:8080/polling/resources/polls/629>;
rel="prev"; type="application/json"; title="Previous poll",
<http://localhost:8080/polling/resources/polls/632>;
rel="next"; type="application/json"; title="Next poll",
<http://localhost:8080/polling/resources/polls>;
rel="collection"; type="application/json"; title="Polls
collection", <http://localhost:8080/polling/resources/polls>;
rel="collection up";  type="application/json"; title="Self
link", <http://localhost:8080/polling/resources/polls/630>;
rel="self"
```

Example: URLs + HTTP Methods

Uniform Resource Locator (URL)	GET	PUT	POST	DELETE
Collection, such as http://api.example.com/comments/	**List the URIs and perhaps other details of the collection's members.**	**Replace the entire collection with another collection.**	**Create a new entry in the collection. The new entry's URI is assigned automatically and is usually returned by the operation.**	**Delete the entire collection.**
Element, such as http://api.example.com/comments/11	**Retrieve a representation of the addressed member of the collection, expressed in an appropriate Internet media type.**	**Replace the addressed member of the collection, or if it does not exist, create it.**	Not generally used. Treat the addressed member as a collection in its own right and create a new entry in it.	**Delete the addressed member of the collection.**

Source: https://en.wikipedia.org/wiki/Representational_state_transfer

Advantages of REST

- Scalability of component interactions – through layering the client server-communication and enabling load-balancing, shared caching, security policy enforcement;
- Generality of interfaces – allowing simplicity, reliability, security and improved visibility by intermediaries, easy configuration, robustness, and greater efficiency by fully utilizing the capabilities of HTTP protocol;
- Independent development and evolution of components, dynamic evolvability of services, without breaking existing clients.
- Fault tolerant, Recoverable, Secure, Loosely coupled

Richardson's Web Maturity Model

According to **Leonard Richardson** [Talk at QCon, 2008 - http://www.crummy.com/writing/speaking/2008-QCon/act3.html]:

- Level 0 – POX: Single URI (XML-RPC, SOAP)
- Level 1 – Resources: Many URIs, Single Verb (URI Tunneling)
- Level 2 – HTTP Verbs: Many URIs, Many Verbs (CRUD – e.g Amazon S3)
- Level 3 – Hypermedia Links Control the Application State = HATEOAS (Hypertext As The Engine Of Application State) === **truely** RESTful Services

Cross-Origin Resource Sharing(CORS)

- Позволява осъществяване на заявки за ресурси към домейни различни от този за извикващия скрипт, като едновременно предостявя възможност на сървъра да прецени към кои скриптове (от кои домейни – Origin) да връща ресурса и какъв тип заявки да разрешава (GET, POST)

- За да се осъществи това, когато заявката е с HTTP метод различен от GET се прави предварителна (preflight) OPTIONS заявка в отговор на която сървъра връща кои методи са достъпни за съответния Origin и съответния ресурс

CORS HTTP Headers - Simple

- HTTP GET request

```
GET /crossDomainResource/ HTTP/1.1
Referer: http://sample.com/crossDomainMashup/
Origin: http://sample.com
```

- HTTP GET response

```
Access-Control-Allow-Origin: http://sample.com
Content-Type: application/xml
```

CORS HTTP HEADERS – POST ...

- HTTP OPTIONS preflight request

```
OPTIONS /crossDomainPOSTResource/ HTTP/1.1
Origin: http://sample.com
Access-Control-Request-Method: POST
Access-Control-Request-Headers: MYHEADER
```

- HTTP response

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: http://sample.com
Access-Control-Allow-Methods: POST, GET, OPTIONS
Access-Control-Allow-Headers: MYHEADER
Access-Control-Max-Age: 864000
```

Thank's for Your Attention!

Trayan Iliev

CEO of IPT – Intellectual Products & Technologies

http://iproduct.org/

http://robolearn.org/

https://github.com/iproduct

https://twitter.com/trayaniliev

https://www.facebook.com/IPT.EACAD

https://plus.google.com/+IproductOrg